Improved Fortran interfaces for PETSc

Martin Diehl

Improved Fortran interfaces for PETSc

Using PETSc from Fortran is possible, but the current tooling does not allow to create all interfaces automatically. Moreover, the existing interfaces can be improved. This holds even after the ongoing work is MR7517 is finished.

Correct variable names

Currently, variable names are often simply `a`, `b`, `c`. I don't know whether this is only the case for manually created interfaces or also holds when creating them automatically. In the worst case, manual interfaces can be updated by hand (or semi-automatically once), but ideally tooling can be improved.

Better diagnostics

Diagnostics from gfortran for PETSc interfaces implies that they are overloaded

```
/home/m/DAMASK/src/grid/grid_mech_FEM.f90:693:85:

  693 |     call MatSetValuesStencil(Jac,24_pPETScInt,row,24_pPetscInt,col,K_ele,x,err_PETSc)
      |                                                                                     1
Error: There is no specific subroutine for the generic ‘matsetvaluesstencil’ at (1)
```

While a single interface to C could give something like

```
/home/m/DAMASK/src/system_routines.f90:142:11:

  142 |   setCWD = setCWD_C(1.0) /= 0_C_INT
      |           1
Error: Type mismatch in argument ‘cwd’ at (1); passed REAL(4) to CHARACTER(1)
```

It is not fully clear why the messages are different, so we cannot promise that it can be fixed. But it is certainly worth trying

Note: The interfaces are:

```
interface MatReorderForNonzeroDiagonal
  subroutine MatReorderForNonzeroDiagonal(a,b,c,d, z)
    import tMat,tIS
    Mat :: a
    PetscReal :: b
    IS :: c
    IS :: d
    PetscErrorCode z
  end subroutine
end interface
```

and

```
interface

  function setCWD_C(cwd) bind(C)
    use, intrinsic :: ISO_C_Binding, only: C_INT, C_CHAR

    implicit none(type,external)
    integer(C_INT) :: setCWD_C
    character(kind=C_CHAR), dimension(*), intent(in) :: cwd
  end function setCWD_C

end interface
```

so maybe the extra interface layer causes the difference.

Use support of modern Fortran for C bindings

`BIND(C)` is currently not used in PETSc. This could simplify some aspects, for example passing of arrays. It has to be seen whether it is feasible, helpful, and wether incomplete implementation for some subroutines would be helpful. Background: Interoperable Subroutines and Functions (gfortran documentation).